(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
13 September 2001 (13.09.2001)

PCT

(10) International Publication Number
WO 01/67250 A2

(51) International Patent Classification[7]: G06F 12/00

(21) International Application Number: PCT/GB01/00930

(22) International Filing Date: 5 March 2001 (05.03.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/522,199 9 March 2000 (09.03.2000) US

(71) Applicant: INTERNATIONAL BUSINESS MACHINES CORPORATION [US/US]; Intellectual Property Law, New Orchard Road, Armonk, NY 10504 (US).

(71) Applicant (for MC only): IBM UNITED KINGDOM LIMITED [GB/GB]; P.O. Box 41, North Harbour, Portsmouth, Hampshire PO6 3AU (GB).

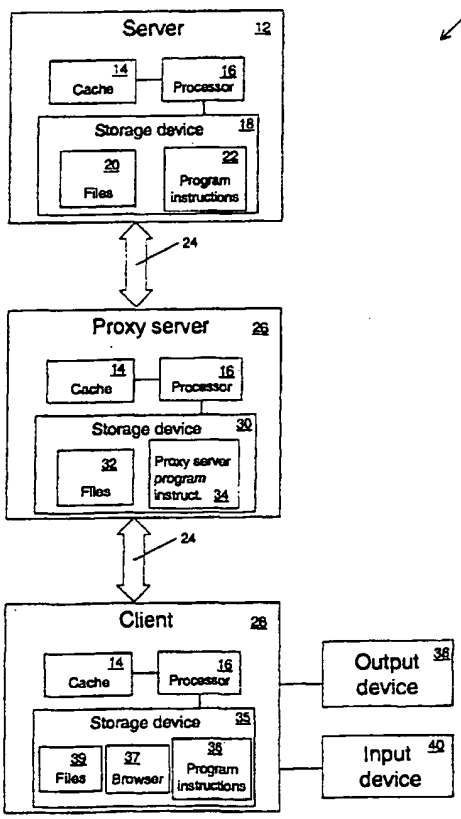
(72) Inventor: DUTTA, Rabindranath; 3401 Palmer Lane W. #835, Austin, TX 78727 (US).

(74) Agent: LING, Christopher, John; IBM United Kingdom Limited, Intellectual Property Law, Hursley Park, Winchester, Hampshire SO21 2JN (GB).

(81) Designated States (national): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

[Continued on next page]

(54) Title: SYSTEM, METHOD AND PROGRAM FOR ORDERED ANTICIPATORY CACHING OF LINKED FILES IN A CLIENT/SERVER NETWORK



(57) Abstract: In a system and method for transferring information using a client/server network, link popularity information characterising the popularity of files linked to a requested file may be transmitted from a server to a client. The popularity may include the relative likelihood of a user request for each of the linked files. The link popularity information may be used in selecting only the more popular linked files for anticipatory caching. The files corresponding to the links most likely to be selected by a user may be loaded into the cache memory of the server and/or downloaded to the client, in advance of any link selection by the user. Ordering of files for anticipatory caching based on their popularity may provide a high probability that the file corresponding to a selected link is already stored on a client computer at the time the link is selected. The time between selection of the link by a user and viewing of the corresponding file may therefore be shortened. Because the most popular linked files are cached first, and less popular files may not be selected for anticipatory caching at all, the time and resources needed to perform the caching may also be reduced as compared to the case of anticipatory caching of all files linked to a requested file.

WO 01/67250 A2

(84) **Designated States** *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations. refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

SYSTEM, METHOD AND PROGRAM FOR ORDERED ANTICIPATORY CACHING OF LINKED FILES IN A CLIENT/SERVER NETWORK

Field of the Invention

This invention relates to information transfer over networks of computational devices, and more particularly to ordered anticipatory caching of files linked to a file requested over a network.

Background of the Invention

The continuing proliferation of powerful, convenient computational devices has been accompanied by an increase in the use of networks connecting these devices. Computational devices include computers and other, often portable, devices such as wireless telephones, personal digital assistants, and automobile-based computers. Such portable computational devices are also sometimes termed "pervasive devices". "Computer", as used herein, may refer to any of such computational devices. The networks connecting computational devices may be "wired" networks, formed using "land lines" such as copper wire or fibre optic cable, wireless networks employing earth and/or satellite-based wireless transmission links, or combinations of wired and wireless network portions. Many networks are organised using a client/server architecture, in which "server" computational devices manage resources, such as files, peripheral devices or processing power, which may be requested by "client" computational devices. The client device is often operated by a user of the network. Computational devices not operated directly by a user, such as "proxy servers" which act on behalf of other machines, may act as either clients or servers.

Currently the most widely used network is the Internet, a global network of computational devices which communicate using a set of protocols called TCP/IP (transmission control protocol/Internet protocol). An especially popular aspect of the Internet is the World Wide Web (WWW, or "web"), a collection of interlinked documents formatted in hypertext markup language (HTML). These documents, or "web pages", may incorporate text, graphics, audio, and/or video content, and may include convenient links to one another, often called "hyperlinks" or simply "links". Documents or files are requested by client computers through an application program called a web browser. The files are requested from server computers, or "web servers". The transmission of the files uses an additional Internet protocol called hypertext transfer protocol (HTTP). A

user viewing a web page on a computer display screen typically "clicks" on a hyperlink using a pointing device, whereupon the linked document is then transferred to the user's computer and displayed.

The file corresponding to a link selected by a user may not appear instantaneously on the user's display screen, however. The linked file is typically stored on a storage device, such as a hard drive, associated with the web server. The file must therefore typically be retrieved from the storage device and then transferred over the Internet to the user's computer. The time taken for this retrieval and transfer can contribute to a noticeable (and sometimes quite annoying) delay between selection of a link and viewing of the associated file or document. Some web servers may provide partial relief of this problem by keeping a requested file in a cache memory for some period of time after the file is requested. A cache memory as used herein refers to a memory allowing rapid access to stored items. Typically, a cache memory is a portion of system memory allocated for short-term storage of frequently used information. Because a file in cache memory may be transmitted without incurring a delay associated with retrieving the file from a long-term storage device, a user requesting a file which has been requested recently (possibly by a different user) and remains in a cache may experience a shorter delay between selecting the corresponding link and viewing the file. Such caching of a file after its request may be implemented, for example, by proxy servers transmitting web documents between web servers and multiple users. Such proxy servers are often used, for example, by large Internet service providers, or ISPs. Although this type of caching may reduce delays associated with viewing certain documents, the probability that a file corresponding to a link selected by a particular user is in a cache memory is not necessarily high.

Another possible approach to lessening the delays associated with viewing of linked files is sometimes called "anticipatory caching". In anticipatory caching, files linked to a requested file (via hyperlinks) are moved to the cache memory of the web server and/or downloaded to the client computer in advance of a request by the user for any of the linked files. A typical web page has multiple linked documents, however, and many of the corresponding links may not be selected by the viewer of the web page. Caching and/or downloading all of the linked documents can therefore waste time and unnecessarily tie up resources of the server and client computers (e.g., cache memory space) and of the network (transmission bandwidth). Unnecessary loading of the network's

transmission capability can in turn increase transmission delays across the network in general.

It would therefore be desirable to develop a system and method to reduce the delays associated with viewing of linked documents such as web pages using a network such as the Internet. The desired method would reduce the delay associated with viewing of a large fraction of selected links, while minimising waste of time and resources.

DISCLOSURE OF THE INVENTION

The problems outlined above are in large part addressed by a system and method for transferring link popularity information between a network server and a network client. The link popularity information characterises the popularity of the files linked to a requested file. The popularity may include the relative likelihood of a user request for each of the linked files. The link popularity information may be generated in various ways, which may include analysing a server log file, tracking "hits" on, or visits to, various web pages, or analysing information content such as "cookies" which may be sent to the server by clients. Transfer of the link popularity information from the server to the client may be done in various ways, also. For example, the information may be stored in a separate file which is sent to the client along with a requested file. The link popularity information could also be incorporated into a protocol, such as the HTTP protocol, used to send the requested file. Furthermore, the link popularity information could be included within the requested file itself, such as by inclusion within the code (e.g. HTML) used to define the links to the linked files.

The link popularity information may be used in selecting only the more popular linked files for anticipatory caching. The files corresponding to the links most likely to be selected by a user may be loaded into the cache memory of the server, and/or downloaded to the client computational device, in advance of any link selection by the user. This ordering of files for anticipatory caching based on their popularity may provide a high probability that the file corresponding to a selected link is already stored on a client computer at the time the link is selected. The time between selection of the link by a user and viewing of the corresponding file (sometimes referred to as "latency") may therefore be greatly shortened. Because the most popular linked files are cached first, and less popular files may not be selected for anticipatory caching at all, the time and resources needed to perform the caching may also be

reduced as compared to the case of anticipatory caching of all files linked to a requested file.

In an embodiment of the method, the link popularity information is transferred from the server to the client at approximately the same time that the requested file containing the links corresponding to the popularity information is transferred to the client. At least one of the linked files may then be transferred from the server to the client, where these linked files are transferred in an order determined using the link popularity information. In a preferred embodiment, this transfer of linked files is a download initiated by the client. Upon selection by a user of a link within the original requested document corresponding to one of the linked files, the linked file may then be displayed on the user's display screen (typically a display screen of the client computational device).

An embodiment of a system for transferring information within a client/server network includes a network server adapted to provide a requested file to a network client, where the network server is further adapted to provide link popularity information associated with the requested file to the network client. The network server may include a processor, a cache memory, a storage device such as a hard drive, and a transmission medium connection. The requested file and additional files linked to the requested file are typically stored on the storage device. The system may further include the network client, where the client is adapted to download one or more of the additional linked files from the server in advance of a user request for any of the linked files. The order of downloading such linked files is determined using the link popularity information. The client is typically adapted to download the additional files using a web browser. The web browser may be adapted to allow a user to choose whether the anticipatory caching is enabled or not.

In addition to the method and system described above, a computer-usable carrier medium is contemplated herein. The carrier medium may be a storage medium, such as a magnetic or optical disk, a magnetic tape, or a memory. In addition, the carrier medium may be a transmission medium, such as a wire, cable, or wireless medium along which data or program instructions are transmitted, or a signal carrying the data or program instructions along such a wire, cable or wireless medium. The carrier medium may contain program instructions executable for carrying out embodiments of the methods described herein.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

Fig. 1 is a block diagram illustrating an embodiment of a system for transferring information within a client/server network;

Fig. 2 illustrates exemplary link popularity data for links within a file accessible using a client/server network;

Fig. 3 is a flow diagram illustrating an embodiment of a method which may be performed by a server for transferring information within a client/server network; and

Fig. 4 is a flow diagram illustrating an embodiment of a method which may be performed by a client for transferring information within a client/server network.

DETAILED DESCRIPTION OF THE INVENTION

Turning now to the drawings, Fig. 1 illustrates an embodiment of a system 10 for transferring information within a client/server network. System 10 includes a network server 12, a computational device which may be a web server. In the embodiment of Fig. 1, server 12 includes cache 14, processor 16 and storage device 18. Cache 14 is a collection of storage locations which are rapidly accessible by processor 16. In an embodiment, cache 14 may be a portion of the processor's system memory allocated for temporary information storage. Storage device, or storage medium, 18 may take many forms, such as volatile or nonvolatile memory, a magnetic disk such as a hard drive or floppy drive, an optical disk, and/or a magnetic tape. Such a storage device is sometimes referred to as a "direct access storage device" (DASD). Typically, storage device 18 has a larger storage capacity than cache 14, but takes longer for processor 16 to access.

In the embodiment of Fig. 1, storage device 18 includes files 20 and program instructions 22, also referred to as program executables. The program instructions are typically stored as "executable files" in a storage device and loaded into memory during execution. Files 20 may include documents such as web pages suitable for viewing by a user of the

network, and may contain text, graphics, video and/or audio information. Such document files may be in the HTML language, or in other suitable languages such as Extensible Markup Language (XML) or Wireless Markup Language (WML). Files 20 may further include data files suitable for use by computational devices in communicating across the network. For example, a file containing link popularity information for the additional files linked to a particular file may be included in files 20. "Files" as used herein may refer to any collection of data suitable for storing on a computational device or transferring within a network. Program instructions 22 may include various program instructions used to implement functions of network server 12, such as program instructions used to implement the methods described herein.

Transmission medium 24 may be used to connect network server 12 to other computational devices, such as proxy server 26 and/or client 28. Transmission medium 24 may include, for example, a wire, cable, wireless transmission path, or a combination of these. Protocols used for transmission along transmission medium 24 may include TCP/IP, HTTP, and/or other suitable protocols such as Wireless Applications Protocol (WAP).

System 10 may include client 28, linked to server 12 using transmission medium 24. In the embodiment of Fig. 1, client 28 includes cache 14, processor 16, and storage device 35. Storage device 35 is similar to storage device 18 described above, and may include files 39, a browser program 37 and program instructions 36. Although a browser program such as browser 37 is implemented using program instructions (or executables) such as instructions 36, browser 37 is shown separately in Fig. 1 to emphasise this feature of the client. Browser program 37 may be, for example, a web browser which allows a user to retrieve and view files on the WWW, or a program which performs a similar function on some other network. In some embodiments, client functions involved in implementation of the methods described herein are included in browser 37. Such functions, and/or other functions of the client computing device, may also be implemented in separate program instructions such as program instructions 36. Files 39 may include various files stored on the client computational device, including files downloaded from a network server such as server 12. Client 28 is typically associated with an output device 38 and input device 40, particularly in embodiments for which the client computational device is operated by a user of system 10. Output device 38 may include, for example, a display screen and/or a printer. Input device 40 may include, for example, a keyboard and/or a pointing device such as a mouse.

In some embodiments, system 10 may also include a proxy server 26. A proxy server as used herein refers to a computational device which acts as an intermediary between a client and a "real" server. The proxy server may appear as a server to the client, and as a client to the real server. Requests from the client may be responded to by the proxy server, or passed on to the real server. Files or other communications from the real server may be passed on to the client by the proxy server. A proxy server may be employed in system 10 for various reasons. For example, proxy servers may be used to provide specialised content and/or improved performance to a selected group of client computers. An example of this may be the use of caching by proxy servers to increase the speed of providing some files to subscribers to a particular ISP. Another use of a proxy server may be to filter the information being sent from the client to a real server, and/or in the opposite direction. For example, a proxy server may be used to implement a "firewall" limiting outside access to computers at a company or other institution. In the embodiment of Fig. 1, proxy server 26 includes cache 14, processor 16, and storage device 30. Storage device 30 may include files 32 and proxy server program instructions 34. Files 32 may include any files stored on the proxy server, such as files being transmitted between a server and a client. Program instructions 34 may include various program instructions used to implement functions of proxy server 26, such as interacting with client computers.

In Fig. 1 and any other block diagrams appearing herein, the blocks are intended to represent functionality rather than specific structure. Implementation of the represented system using circuitry and/or software could involve combination of multiple blocks into a single circuit or device, or combination of multiple circuits and/or devices to realise the function of a block. For example, cache 14 may be included on a semiconductor chip embodying processor 16. Furthermore, a system such as system 10 may include other elements not explicitly shown. For example, multiple servers, proxy servers, and/or clients not shown in Fig. 1 may be included in a system used for implementing the methods described herein. Further, the server, proxy server, and/or client computational devices may themselves include additional elements not shown.

An exemplary set of link popularity information 41 which may be used in implementing the methods described herein is shown in Fig. 2. Requested file 42 is a file which may be requested by a user (through, for example, a web browser), and accordingly requested from a server by a client. In the data of Fig. 2, the requested file is an HTML file called

"corporationx.html". Linked files 44 are files which are linked to within the corresponding requested file 42. In the embodiment of Fig. 2, requested file "corporationx.html" includes links to seven linked files, including "products.html" and "address.html". In embodiments for which requested file 42 is an HTML file, links to linked files 44 may be included in requested file 42 using HTML formatting tags which define the links. Each of linked files 44 is associated with a corresponding popularity indicator 46. In the embodiment of Fig. 2, each of popularity indicators 46 is expressed as a probability that the corresponding linked file will be chosen by a user selecting a link within requested file 42. Since in this example the indicators for all of the linked files add up to one, forty percent of the link selections from "corporationx.html" are likely to be selections of the link corresponding to "products.html". Popularity indicators 46 may be expressed in various other forms, however. For example, a percentage of link selections which select the corresponding file could be stated for each file, or a percentage of users viewing the selected file who select the corresponding link could be used. As another alternative, an actual number of selections of the link over a certain time period could be given, or possibly a selection rate.

In view of link popularity information 41 of Fig. 2, the server containing requested file 42 and linked files 44 (typically on a storage device such as device 18 of Fig. 1) may, upon transferring requested file "corporationx.html" to a client, also load linked files "products.html" and "support.html" into a cache memory on the server, such as cache 14 of Fig. 1. If either of these linked files are subsequently requested by the client, they can therefore be transferred rapidly to the client, without a delay associated with reading the files from the storage device. Link popularity information 41 may also be made available to the client. In view of the link popularity information, the client may download, for example, linked files "products.html" and "support.html" from the server in advance of any selection by a user of a linked file. In this way, if either of these linked files are selected by the user, the file will already be on the client computer, and delay associated with transmission between the server and the client will be eliminated. Because link popularity information 41 indicates that about seventy percent of link selections are for either "products.html" or "support.html", caching and/or downloading of these two files should reduce latency for about seventy percent of link selections from "corporationx.html". This performance improvement may be achieved without the need to use additional time and resources by caching and/or downloading the other, less popular linked files.

Determination of popularity indicators 46 to form link popularity information 41 may be accomplished in various ways. For example, a tracking program may be used to tabulate the number of times particular files stored by a web server are accessed. Multiple implementations of such tracking programs, which typically provide additional information such as which specific users are accessing the files, are currently commercially available. In some embodiments "cookies", or identifying information which may be sent to a web server by a web browser, may be used for tracking selections of particular linked files. An alternative way of determining popularity of linked files is to analyse a server log, or a file containing a record of activity on the server. Web servers typically maintain such log files including each request made to the server. In another alternative embodiment, link popularity information may be estimated based on, for example, knowledge of the nature of the links. Such an estimate may in some embodiments be used as an initial or default setting of the link popularity information, which could be subsequently updated based upon, e.g. server statistics or log file analysis. Updates of the link popularity information may in some embodiments be performed regularly, possibly in conjunction with other updates and maintenance associated with the server.

Link popularity information such as information 41, which is shown in tabular form in Fig. 2, may be stored on a server and transferred to a client in multiple different ways. In one embodiment, information similar to that in Fig. 2 may be stored in a separate file, which may be stored on the server on a device such as storage device 18 of Fig. 1. Such a file, which in the embodiment of Fig. 2 might have a name such as "corporationx_linkpopularity.html", could be transferred to a client at the same time the requested file ("corporationx.html") is transferred. The file could include various programming languages, and any of multiple data structures (e.g., tabular structures or object-oriented structures) could be used to relate the information. Alternatively, link popularity information could be included within the requested file itself, such as within the formatting used to establish each link to a linked file. In such an embodiment, the client browser program is preferably configured to extract the link popularity information from the requested file. The link popularity information might also be included in a header to the requested file, a portion typically at the beginning of the file reserved for information regarding the file. As another example, the link popularity information might be sent from the server to the client using a communications protocol such as HTTP (e.g., in its header information). In some embodiments, implementation of methods for sending link popularity

information may involve enhancement of the specifications for programming languages (such as HTML) or communications protocols (such as HTTP). The rapid evolution of computer-based networks typically results in frequent enhancements and upgrades to such specifications.

The link popularity information shown in Fig. 2 represents merely an exemplary embodiment, and many other embodiments are possible and contemplated. For example, the data of Fig. 2 could include additional possible requested files 42, along with corresponding linked files and popularity indicators. In fact, any of the linked files 44 shown in Fig. 2 could also contain linked files themselves, and as such be included as one of requested files 42. The link popularity information may also be arranged in forms other than that shown in Fig. 2. For example, a requested file could be represented as an object in an object-oriented programming approach, and the linked files and corresponding popularity indicators could be represented as attributes of such an object. Furthermore, other issues not explicitly illustrated in Fig. 2 could be included in the link popularity information. For example, in addition to characterising the likelihood of a linked file being selected for viewing, the link popularity information could indicate likelihood, for example, of a link being the first link selected.

Turning now to Fig. 3, a flow diagram illustrating an embodiment of a method for transferring information within a client/server network is shown. The method of Fig. 3 may be performed by a server such as server 12 of Fig. 1, and may be implemented using program instructions such as instructions 22. In the embodiment of Fig. 3, after a request is received from a client for a file stored on the server (box 48), link popularity information for the requested file is compiled or located (box 50). The client requesting the file may be, for example, a client such as client 28 of Fig. 1, or a proxy server such as proxy server 26 of Fig. 1. In some embodiments, the link popularity information is already stored on the server at the time the request for a file is received. As discussed above in the description of Fig. 2, the link popularity information may be included, for example, in a separate file or within the requested file itself. Using link popularity information which is already stored may improve the speed with which the server responds to the request for a file. Alternatively, the link popularity information may be compiled in response to the file request in some embodiments, so that the most up-to-date popularity information may be provided. In another possible embodiment, the link popularity information for the requested file could be updated at the time of the request by accessing a separate file

containing link popularity information which is regularly updated. Such a method may be useful particularly in embodiments for which the link popularity information is transmitted within the requested file itself or via a communications protocol.

The requested file is transmitted to the requesting computational device (box 52), and the link popularity information is also transmitted (box 54). As discussed above, the link popularity information may be sent in various ways, including as a file, within the requested file, and/or within transmission protocol commands or headers. In the embodiment of Fig. 3, one or more of the linked files are loaded into a cache memory on the server according to the relative popularity of the linked files (box 56). Various criteria may be used to determine how many files are cached, and in some embodiments a user (typically a server administrator) may be able to modify the criteria. For example, linked files could be cached in order of popularity until enough files are cached that a predetermined percentage of link selections is likely to be met using the cached files. Using the exemplary information of Fig. 2, if criteria were established dictating that enough files would be cached to meet seventy percent of likely link selections from the cached files, then caching of "products.html" and "support.html" would be sufficient. If, however, the criteria dictated that seventy-five percent of likely link selections should be met using the cached files, then "stockholder.html" would also be cached. Many other types of criteria could also be used, such as caching the three most popular files, or caching files in order of popularity until a predetermined amount of time has elapsed. In cases for which multiple files have the same probability of being selected by a user, other types of link popularity information may also be available for use in determining a caching order. For example, the link popularity information may in some embodiments include a probability that a file is selected first, in addition to a probability that a file is selected at all. The cached files are typically removed from the cache (or overwritten by other information) after some predetermined time has elapsed.

In the embodiment of Fig. 3, one or more of the cached linked files are sent to the client (box 58). This is typically done in response to a request from the client, through which the client may cache the most popular linked files in anticipation of a user request for such files. If the linked files themselves contain links to other files, then the sequence of steps 48 through 58 could be carried out for the requested linked files. This may in some cases consume excessive time and

resources, however, particularly when it is not known whether the link corresponding to a requested linked file will actually be selected by a user. In some embodiments, therefore, the request from a client for a file may include information as to whether the file is user-requested or requested for anticipatory caching (in advance of any user request). In such an embodiment, the server may be adapted to send link popularity information only for user-requested files.

Many variations of the method of Fig. 3 are possible and contemplated. For example, if a requested file contains no links, then steps related to linked files would naturally be omitted. The methods described herein are believed to reduce the time and resources spent in providing anticipatory caching as compared to methods in which all linked files are cached. There is some cost in time and resources even for these methods, however, and in some cases a user may wish to disable the ordered anticipatory caching. Disabling of the caching might also be preferred, for example, in situations for which there is a possibility of interference with other applications which may make use of caching. Such disabling could be implemented using, for example, an option in a browser program. In such an embodiment, a request from a client for a file could include, for example, instructions not to send link popularity information. Steps 50 and 54-58 could be omitted in such a case. Some of the steps could also be performed in a different order than shown in Fig. 3. For example, the order of steps 50 and 52 could be reversed in some embodiments, as well as the order of steps 54 and 56.

A flow diagram illustrating another embodiment of a method for transferring information within a client/server network is shown in Fig. 4. The method of Fig. 4 may be performed by a client such as client 28 of Fig. 1, or by a computer acting as a client, such as a proxy server. The method is typically implemented on a client using a browser program such as browser 37 of Fig. 1, but some or all of the method could also be implemented using other program instructions, such as instructions 36 or proxy server instructions 34. In the embodiment of Fig. 4, a file is requested from a server (or possibly a proxy server) in response to a user selection of the file (box 60). As noted above in the description of Fig. 3, such a request to the server may in some embodiments include notification that the request is in response to a user selection (as opposed to a request of a file for anticipatory caching). The request may also include other information in some embodiments, such as a notification that link popularity information is not to be provided (in an embodiment for which ordered anticipatory caching is disabled by a user). The

requested file is received from the server (box 62), and in embodiments for which the ordered anticipatory caching is not disabled, link popularity information for any linked files within the requested file is received as well (box 64). The requested file is displayed on the user's display screen (box 66).

Based on the received link popularity information, one or more of the linked files associated with the requested file may be requested from the server for anticipatory downloading (box 68). The criteria used to determine how many linked files are requested from the server for downloading may be similar to those used in selecting files for caching on the server, discussed above in the description of Fig. 3. The requested linked files are then received from the server, and may be stored in a cache memory of the client computer (box 70). If the next file requested by the user (e.g. by selecting a link) is one of these cached linked files ("yes" branch of decision box 72), the requested file may be rapidly displayed (box 74). If the requested file is not one of the cached linked files ("no" branch of 72), the file is requested from the server (box 60). If the new requested file has linked files within it, steps 62 through 70 may be repeated for the new requested file. Multiple variations of the method of Fig. 4 are possible and contemplated. As in the case of the method of Fig. 3, for example, steps relating to linked files may be omitted in embodiments for which the requested file contains no links, or the ordered anticipatory caching is disabled by a user. Some of the steps in Fig. 4 could be performed in a different order without affecting the usefulness of the method. For example, the ordering of steps 62, 64 and 66 could be changed in some embodiments.

Program instructions, such as instructions 22, 34 or 36 of Fig. 1 or instructions within browser 37, implementing methods such as those illustrated by Figs. 3 and 4 may be transmitted over or stored on a carrier medium. The carrier medium may be a transmission medium such as a wire, cable, or wireless transmission link, or a signal travelling along such a wire, cable or link. The carrier medium may also be a storage medium, such as a volatile or non-volatile memory (e.g., read-only memory or random access memory), a magnetic or optical disk, or a magnetic tape.

CLAIMS

1. A method of transferring information between a network server and a network client, said method comprising:

transferring a requested file from the server to the client; and

transferring link popularity information associated with the requested file from the server to the client, wherein the link popularity information characterises a relative likelihood of user requests for access to each of multiple additional files linked to the requested file.

2. The method as recited in claim 1, wherein said transferring a requested file comprises transferring a web page file.

3. The method as recited in claim 1, wherein said transferring a requested file comprises transferring a markup language document.

4. The method as recited in claim 1, wherein said transferring link popularity information comprises embedding the link popularity information into the requested file.

5. The method as recited in claim 1, further comprising transferring at least one of said multiple additional files from the server to the client, wherein said additional files are transferred in an order determined using the link popularity information, and wherein the additional files are transferred prior to a user request for any of the additional files.

6. The method as recited in claim 5, wherein said transferring at least one of the additional files comprises transferring the at least one of the additional files to a cache memory associated with the client.

7. The method as recited in claim 5, further comprising loading the at least one of said additional files into a cache memory associated with the server, prior to said transferring the at least one of said additional files.

8. The method as recited in claim 5, further comprising displaying one or more of said additional files in response to a user request.

9. The method as recited in claim 1, further comprising compiling said link popularity information, prior to said transferring link popularity information.

10. The method as recited in claim 9, wherein said compiling link popularity information comprises analysing a server log file.

11. The method as recited in claim 5, wherein said transferring at least one of said multiple additional files is done in response to a request from the client to the server.

12. The method as recited in claim 7, wherein said transferring at least one of said multiple additional files is done in response to a request from the client to the server, and wherein said loading the at least one of the additional files into a cache memory associated with the server is done prior to the request from the client to the server.

13. A method of obtaining information using a client/server network, said method comprising:

receiving a requested file from a network server;

receiving link popularity information from the network server, wherein the link popularity information characterises a relative likelihood of user requests for access to each of multiple additional files linked to the requested file;

downloading at least one of the multiple additional files into a memory, wherein said at least one of the additional files are downloaded in an order determined using the link popularity information, and wherein said downloading is performed prior to any user request for the additional files; and

displaying one or more of the downloaded additional files, responsive to a user request.

14. The method as recited in claim 13, wherein said receiving link popularity information comprises receiving a file containing the link popularity information.

15. The method as recited in claim 13, wherein said receiving link popularity information comprises receiving information sent using transmission protocol headers.

16. The method as recited in claim 13, wherein said receiving link popularity information comprises receiving information embedded in the requested file.

17. A system for transferring information within a client/server network, said system comprising:

a network server adapted to provide a requested file to a network client, wherein the network server is further adapted to provide link popularity information associated with the requested file to the network client, and wherein the link popularity information characterises a relative likelihood of user requests for access to each of multiple additional files linked to the requested file.

18. The system as recited in claim 17, wherein the network server comprises a computational device.

19. The system as recited in claim 18, wherein the computational device comprises a computer, telephone or personal digital assistant.

20. The system as recited in claim 17, wherein the network server comprises a processor, a cache memory, a storage device, and a transmission medium connection.

21. The system as recited in claim 20, wherein the requested file and the multiple linked additional files are stored upon the storage device, and wherein the additional files are linked to the file using links embedded within the file.

22. The system as recited in claim 21, wherein said requested file comprises a web page file, and said links comprise hyperlinks.

23. The system as recited in claim 17, further comprising the network client, wherein the network client is adapted to download one or more of the additional linked files from the network server in advance of a user request for the one or more files, and wherein the order of downloading the one or more additional files is determined using the link popularity information.

24. The system as recited in claim 23, wherein the network client comprises a computational device.

25. The system as recited in claim 24, wherein the computational device comprises a computer, telephone or personal digital assistant.

26. The system as recited in claim 23, wherein the network client comprises a proxy device through which information is transmitted between the network server and an additional network client.

27. The system as recited in claim 23, wherein said network client comprises a web browser program.

28. The system as recited in claim 27, wherein said web browser program is adapted to allow said downloading of the additional linked files in advance of a user request to be blocked by the user.

29. A computer-usable carrier medium, comprising:

first program instructions executable on a computational device for transferring link popularity information associated with a requested file from the computational device to a client computational device, wherein the link popularity information characterises a relative likelihood of user requests for access to each of multiple additional files linked to the requested file.

30. The carrier medium as recited in claim 29, further comprising second program instructions executable on the computational device for transferring the requested file from the computational device to the client computational device.

31. The carrier medium as recited in claim 30, further comprising third program instructions executable on the computational device for transferring at least one of said multiple additional files from the computational device to the client computational device, wherein the at least one of the additional files is transferred in an order determined using the link popularity information.

32. The carrier medium as recited in claim 31, further comprising fourth program instructions executable for compiling said link priority information.

33. A computer-usable carrier medium, comprising:

first program instructions executable on a computational device for receiving a requested file from a server computational device;

second program instructions executable on the computational device for receiving link popularity information from the server computational device, wherein the link popularity information characterises a relative likelihood of user requests for access to each of multiple additional files linked to the requested file; and

third program instructions executable on the computational device for downloading at least one of the multiple additional files from the server computational device into a memory on the computational device,

wherein said at least one of the additional files are downloaded in an order determined using the link popularity information, and wherein the downloading is performed prior to any user request for the additional files.

34. The carrier medium as recited in claim 33, further comprising fourth program instructions executable on the computational device for displaying one or more of the downloaded additional files in response to a user request.

35. A method of transferring information between a network server and a network client, said method comprising:

transferring a requested file from the server to the client; and

transferring one or more of multiple additional files linked to the requested file from the server to the client, wherein the additional files are transferred in an order determined using link popularity information associated with the requested file, and wherein the link popularity information characterises a relative likelihood of user requests for access to each of multiple additional files linked to the requested file.

36. The method as recited in claim 35, further comprising loading the one or more of the additional files into a cache memory associated with the server, prior to said transferring one or more of the multiple additional files.

37. The method as recited in claim 35, wherein said transferring one or more of the multiple additional files is performed prior to any user request for the additional files.

38. The method as recited in claim 35, wherein said transferring one or more of the multiple additional files is done in response to a request from the client to the server.

39. The method as recited in claim 35, further comprising compiling the link popularity information, prior to said transferring one or more of the multiple additional files.

40. A method of transferring information between a network server and a network client, said method comprising:

transferring a requested file from the server to the client; and

loading one or more of multiple additional files linked to the requested file into a cache memory associated with the server, wherein the

additional files are loaded in an order determined using link popularity information associated with the requested file, and wherein the link popularity information characterises a relative likelihood of user requests for access to each of multiple additional files linked to the requested file.

41. The method as recited in claim 40, wherein said loading is done prior to any request from the client to the server for said multiple additional files.

42. The method as recited in claim 40, further comprising compiling said link popularity data, prior to said loading.

43. A method of obtaining information using a client/server network, said method comprising:

receiving a requested file from a network server; and

downloading at least one of multiple additional files linked to the requested file into a memory, wherein said at least one of the additional files are downloaded in an order determined using link popularity information associated with the requested file, and wherein the link popularity information characterises a relative likelihood of user requests for access to each of multiple additional files.

44. The method as recited in claim 43, wherein said downloading is performed prior to any user request for the additional files.

45. The method as recited in claim 43, further comprising displaying one or more of the downloaded additional files, responsive to a user request.

46. The method as recited in claim 43, further comprising receiving the link popularity information from the network server, prior to said downloading.

47. The method as recited in claim 46, wherein said receiving the link popularity information comprises receiving a file containing the information.

48. The method as recited in claim 46, wherein said receiving the link popularity information comprises receiving information sent using transmission protocol headers.

49. The method as recited in claim 46, wherein said receiving the link popularity information comprises receiving information embedded in the requested file.

50. A system for transferring information within a client/server network, said system comprising:

a network server;

a file stored on the network server; and

a means for associating link popularity information with the file, wherein the link popularity information characterises a relative likelihood of user requests for access to each of multiple additional files linked to the file.

51. The system as recited in claim 50, wherein said means for associating comprises an additional file containing the link popularity information.

52. The system as recited in claim 50, wherein said means for associating comprises a network communications protocol.

53. The system as recited in claim 50, further comprising:

a cache memory within the network server; and

a means for loading one or more of the multiple additional files into the cache memory in an order determined using the link popularity information.

54. The system as recited in claim 53, wherein the means for loading comprises a means for loading the one or more multiple additional files into the cache memory prior to any request from a network client for the additional files.

55. The system as recited in claim 50, further comprising:

a means for sending the file to a network client; and

a means for sending one or more of the multiple additional files to the network client, wherein the one or more multiple additional files are sent in an order determined using the link popularity information.

56. The system as recited in claim 55, wherein said means for sending the file comprises a means for sending the file in response to a request from the client, and wherein said means for sending one or more of the multiple additional files comprises a means for sending the one or more additional files in response to a request from the client.

57. The system as recited in claim 55, further comprising the network client.

58. A computer-usable carrier medium, comprising:

first program instructions executable on a computational device for transferring to a client computational device one or more multiple additional files linked to a requested file, wherein the additional files are transferred in an order determined using link popularity information associated with the requested file, and wherein the link popularity information characterises a relative likelihood of user requests for access to each of the multiple additional files.

59. The carrier medium as recited in claim 58, further comprising second program instructions executable on the computational device for transferring the requested file to the client computational device, prior to said transferring the one or more additional files.

60. The carrier medium as recited in claim 58, wherein said first program instructions are further executable for transferring the one or more additional files prior to a user request for said additional files.

61. The carrier medium as recited in claim 58, further comprising second program instructions executable for loading the one or more additional files into a cache memory in an order determined using the link popularity information, prior to said transferring the one or more additional files.

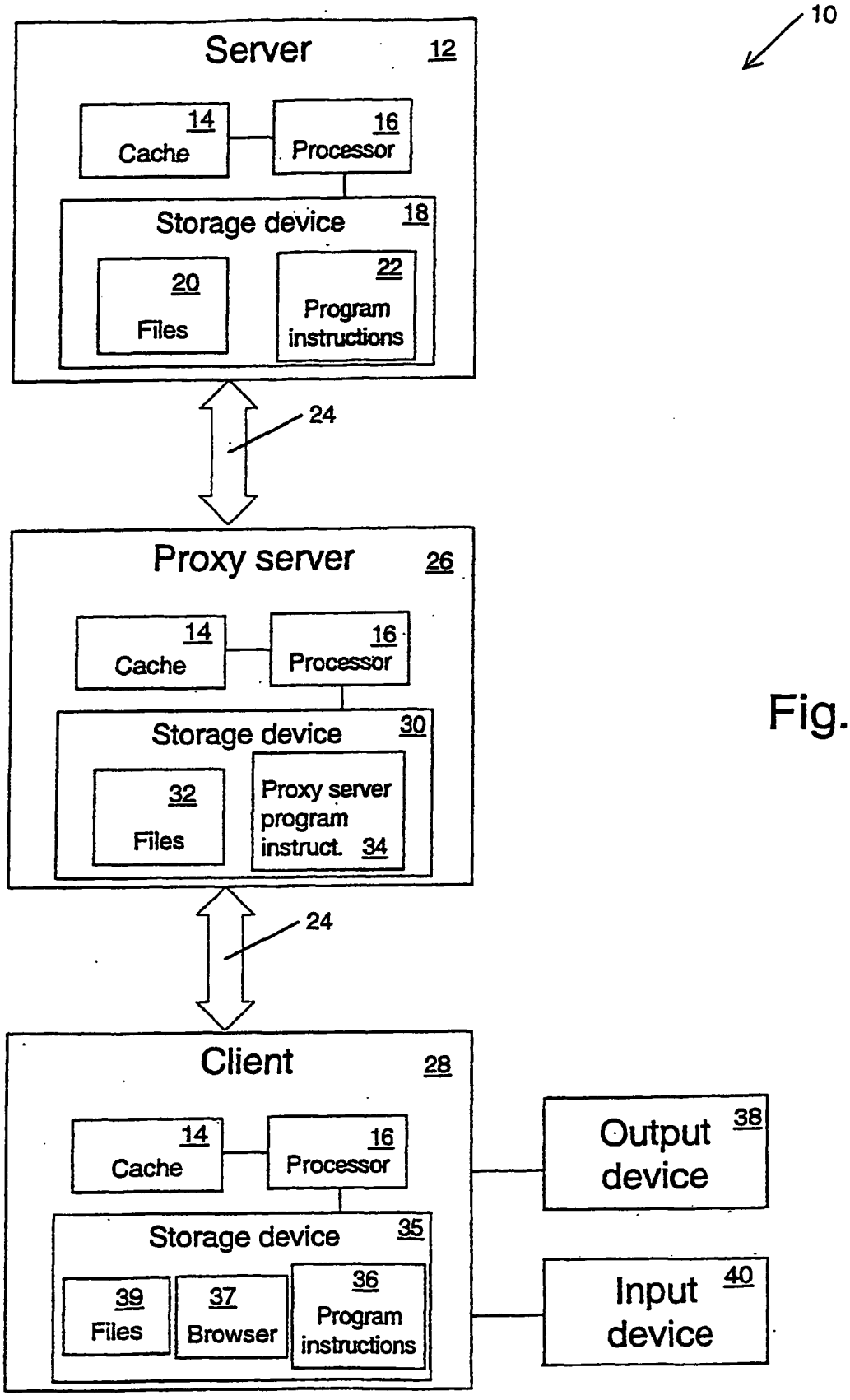

10
Server
12
14
Cache
16
Processor
Storage device
18
20
Files
22
Program instructions
24
Proxy server
26
14
Cache
16
Processor
Storage device
30
32
Files
Proxy server program instruct.
34
24
Client
28
14
Cache
16
Processor
Storage device
35
39
Files
37
Browser
36
Program instructions
Output device
38
Input device
40

Fig. 1

41

42 Requested file	44 Linked file	46 Popularity
corporationx.html	products.html	0.4
	stockholder.html	0.1
	support.html	0.3
	employment.html	0.05
	address.html	0.05
	send-electronic-mail.html	0.05
	history_of_corporationx.html	0.05

Fig. 2

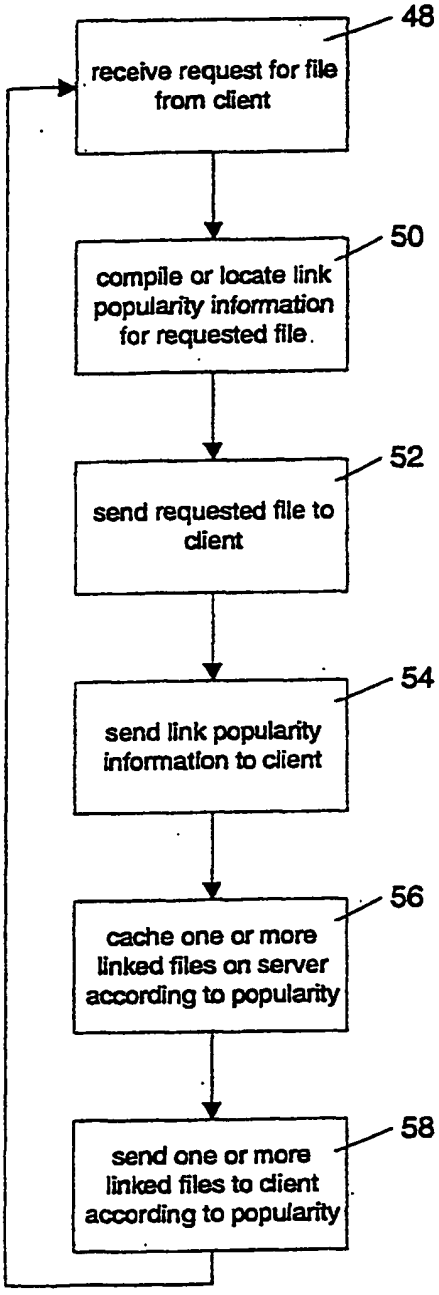

receive request for file from client
48
compile or locate link popularity information for requested file.
50
send requested file to client
52
send link popularity information to client
54
cache one or more linked files on server according to popularity
56
send one or more linked files to client according to popularity
58

Fig. 3

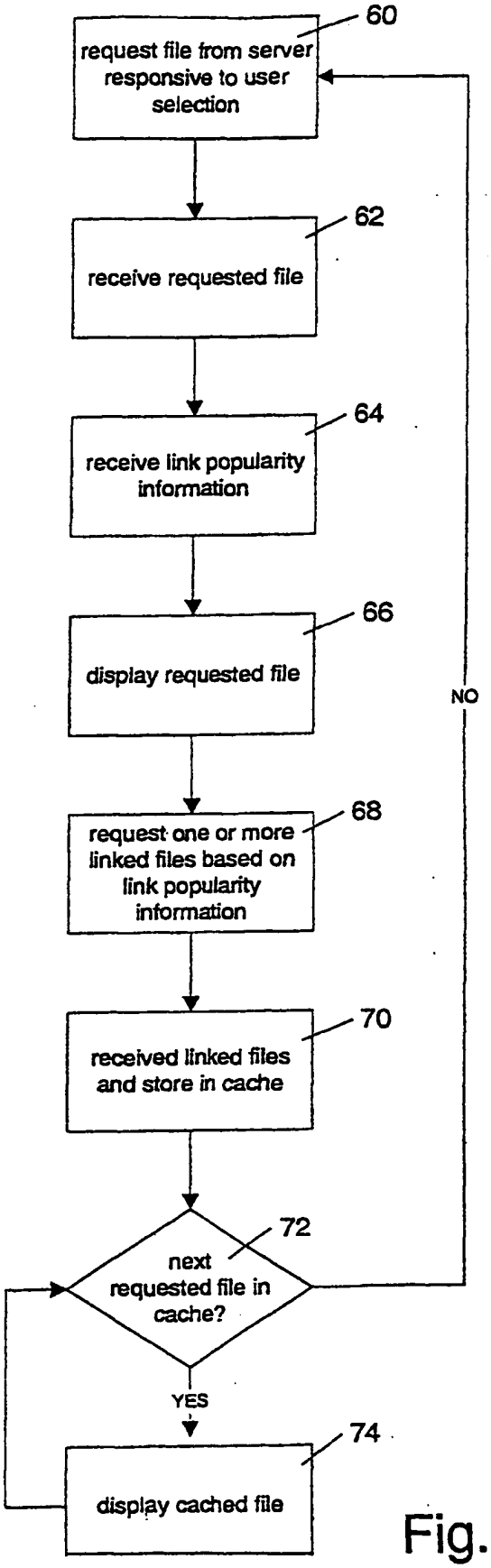

request file from server responsive to user selection
60
receive requested file
62
receive link popularity information
64
display requested file
66
request one or more linked files based on link popularity information
68
received linked files and store in cache
70
next requested file in cache?
72
NO
YES
display cached file
74

Fig. 4